Movie Recommendation

Hsiang-Tai Huang

Wednesday, September 9, 2015

Executive Summary

I use the movie rating dataset provided by GroupLens Research to evaluate the performace of two prediction models which includes user-Based recommendation system and item-based recommendation system. Based on the evaluation resulf of the MovieLens 100k dataset, the precison/recall of user-based recommendation is better than item-based recommendation. When the data is getting better, user-based recommendation will need more memory to process and will consume more time to predict rate. In the such case, item-based recommendation system will would be better than user-based recommendation system.

Preprocessing data

At first, I try to download training dataset and testing dataset from the GroupLens Research website. The ua.base file stands for training dataset and ua.test stand for testing data. Those two datsets includes userid, movieid, rating, and timestamp column.

```
rm(list = ls())
#load ua.base dataset
rating.header <- c('userid','movieid','rating','timestamp')
train.ratings <- do.call(rbind, strsplit(readLines("./ml-100k/ua.base"), '\t', fixed=T))
train.ratings <- data.frame(apply(train.ratings, 2, as.integer))
colnames(train.ratings) <- rating.header

#load ua.test dataset
test.ratings <- do.call(rbind, strsplit(readLines("./ml-100k/ua.test"), '\t', fixed=T))
test.ratings <- data.frame(apply(test.ratings, 2, as.integer))
colnames(test.ratings) <- rating.header
```

After loading data into data frame, I also check the missing values in the training dataset and test dataset.

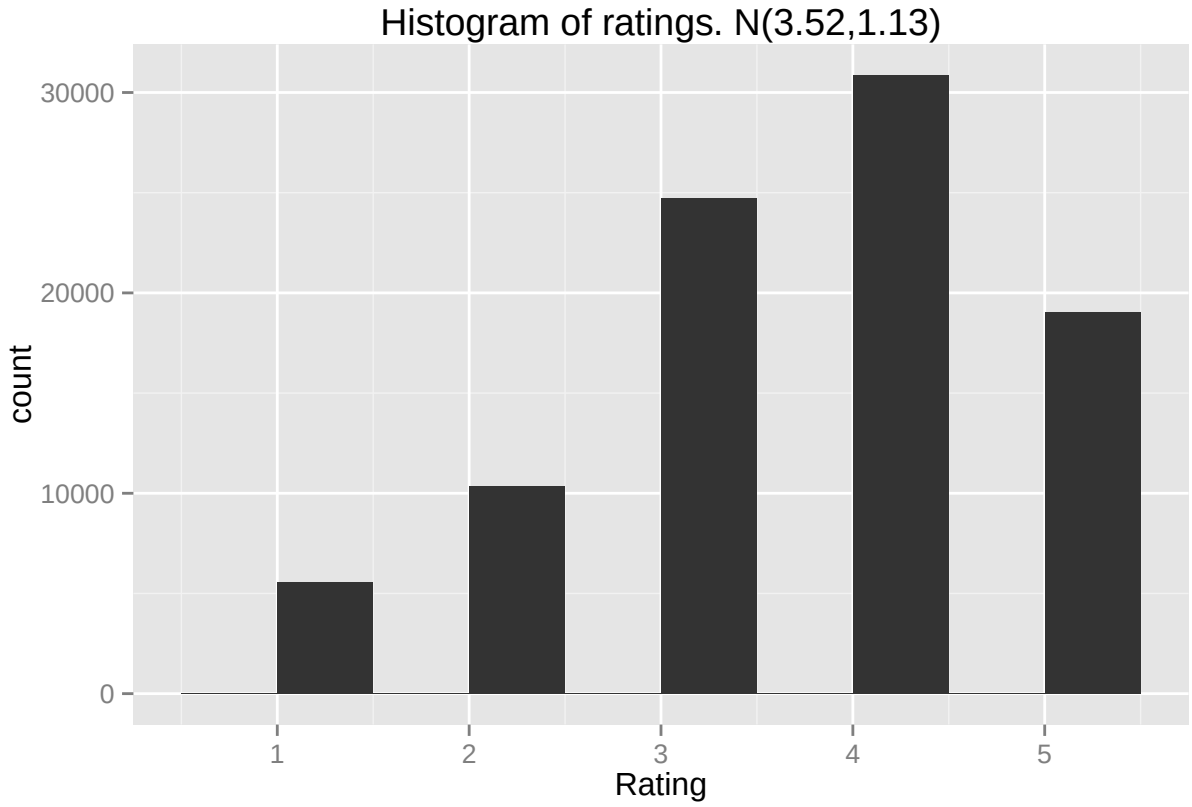
```
train.na.num <-sum(is.na(train.ratings))
test.na.num <-sum(is.na(test.ratings))
```

The na amount in training dataset is 0 and the na amount in test dataset is 0. I don't need to take further step to clean the dataset.

Explore data in training dataset

After cleaning the data, I start to explore the data and draw a plot to show the histogram of rating as the following.

```
library(ggplot2)
mean.rate <-mean(train.ratings$rating)
sd.rate <- sd(train.ratings$rating)
qplot(rating, data = train.ratings ,main = paste0("Histogram of ratings. N(",
  round(mean.rate,2), ",",round(sd.rate,2),")"),binwidth= 0.5, xlab = "Rating")
```



The mean rate would be 3.5238269. This value would be used when we don't have any hint to predict user's rating at some specific situation.

Build recommendation system

I will build up two recommendation systems including user-based recommendation system and item-based recommendation system. Before that, I will initalize the common parameters,matrices and functions. users.movies.mtx would be the most important matrix in this report. Each row of the users.movies.mtx stands for one userid, and each column of the users.movies.mtx stands for one movieid. The rating value which is rated by the User i for the movie j is stored in [i,j]. Besides that, rmse will be used to evalute the prediction accuracy. Finally, I create a matrix named comparion to record computation time,rmse,precision and recall for the two recommendation system.

```
library(knitr)
library(dplyr)
recommend.num <- 5
comparison <- data.frame(rep(0,4),rep(0,4))
names(comparison) <- c("user.based", "item.based")
row.names(comparison) <- c("prcoess.time", "rmse", "precision", "recall")
train.users <- sort(unique(train.ratings[, rating.header[1]]))
train.movies <- sort(unique(train.ratings[, rating.header[2]]))
# build up users.movies matrix
users.movies.mtx <- matrix(NA, nrow = length(train.users), ncol = length(train.movies))
update.user.movies<- function (x) {
  user.idx <- which((train.users %in% x[1]))
```

```
  movie.idx <- which((train.movies %in% x[2]))
  users.movies.mtx[user.idx, movie.idx] <<- x[3]
}
result <- apply(train.ratings, 1, update.user.movies)

rmse <- function(right, predict) {
  sqrt(sum((right - predict)^2)/length(right))
}
```

User-based recommendation system

In the user-based recommendation system, I have set up the user similarity function. This user similarity function would generate the output value which define how closer for two different user. I choose Cosine-based Similarity as my similarity function. After that, I would build up the user similarity matrix. Assumed I have m users and the user similarity matrix would be mxm and store the similarity cefficient in this matrix.

```
#set similarity function
user.sim.func <- function(i, j) {
  (sum(users.movies.mtx[i,] * users.movies.mtx[j,],na.rm =TRUE)) /
    (sqrt(sum(users.movies.mtx[i,]^2,na.rm = TRUE)) *
     sqrt(sum(users.movies.mtx[j,]^2, na.rm = TRUE)))
}
#build up the user similarity matrix
user.similarity <- matrix(NA, nrow=length(train.users), ncol=length(train.users))
for(i in 1:length(train.users)){
        for(j in i:length(train.users)){
          if( i != j){
            user.similarity[i, j] <- do.call(user.sim.func, list(i = i, j = j))
            user.similarity[j, i] <- user.similarity[i, j]
          }
        }
}
```

Item-based recommendation system

In the item-based recommendation system, I have set up the item similarity function. This item similarity function would generate the output value which define how closer for two different item based on user's rating. I choose Cosine-based Similarity as my similarity function. I will record k(500) items which has highest similarity value for each item.

```
#set item similarity function
item.sim.func <- function(i, j) {
  (sum(users.movies.mtx[,i] *users.movies.mtx[,j], na.rm = TRUE))/
    (sqrt(sum(users.movies.mtx[,i]^2, na.rm = TRUE)) *
       sqrt(sum(users.movies.mtx[,j]^2, na.rm = TRUE)))
  }

#build up item similarity list and only record k highest correlation
max.k.value <- 500
item.similarity <- as.list(rep(NA,length(train.movies)))
# get K highest correlation
k.highest <- function(origin, new.data, k) {
```

```
  output <- rbind(origin,new.data)
  if(dim(output)[1] > k) {
    output<-output[order(output[, 1], decreasing = TRUE),]
    output<-head(output,k)
    }
  output
  }

for(i in 1:length(train.movies)){
  for(j in i:length(train.movies)){
    if( i != j){
      sim <- item.sim.func(i, j)
      item.similarity[[i]] <- k.highest(item.similarity[[i]], c(sim, j), max.k.value)
      item.similarity[[j]] <- k.highest(item.similarity[[j]], c(sim, i), max.k.value)
      }
    }
  }
```

Predict and evaluate test dataset

I use the test data set to evaluate those two recommendation system. I also set the total amount of recommended movie is 5.

Evaluate user-based recommendation

I define the function named user.predict.rate. Given userid,moiveid and recommended movie amount(k), I will calculate the k users who had top-5 high similairity with given userid and also rate for this movieid. I will weight rating score and get the predicted rating values. In some special case, I could not get the predicted rating value, I adopt the average rating value from the training dataset as the default rating score.

```
user.predict.rate <- function(uid, mid, k) {
  # Find the user's k nearest neighbour who have already rated movies
  user.rate <- data.frame(user.similarity[uid, ], users.movies.mtx[, mid])
  user.rate <- user.rate[complete.cases(user.rate),]
  user.rate <- user.rate[order(user.rate[, 1], decreasing = TRUE),]
  if(dim(user.rate)[1] == 0) {
    mean.rate
  } else {
    top.k <- head(user.rate, k)
    if(sum(abs(top.k[,1])) == 0) {
      mean(top.k[,2])
    } else {
      sum(top.k[,1] * top.k[,2])/sum(abs(top.k[,1]))
    }
  }
}
start_time <- proc.time()
user.predict.out<- apply(test.ratings, 1, function(x) {round(user.predict.rate(x[1],x[2],recommend.num)
comparison[1,1] <-(proc.time() - start_time)[3]
comparison[2,1] <- rmse(test.ratings$rating, user.predict.out)
user.predict.out<-data.frame(test.ratings[,-c(4)],predict = user.predict.out)
user.precision <-user.predict.out %>% group_by(userid)%>% arrange(desc(predict)) %>% slice(1:recommend.
```

```
user.recall<-user.predict.out %>% group_by(userid)%>%summarise(total.num = sum(rating>=4,na.rm = TRUE))
comparison[3,1] <- mean(user.precision$good/user.precision$recommend.num, na.rm = TRUE)
comparison[4,1] <- mean(user.precision$good/user.recall$total.num, na.rm = TRUE)
```

The (process.time, rmse,precison and recall) for the user-based recommendation system is (5.07, 1.08, 0.68, 0.61)

Evaluate item-based recommendation

I define the function named item.predict.rate. Given userid,moiveid and recommended movie amount(k), I will calculate the K movies which had top-5 high similairity with given movie and also are rated by this userid. I will weight rating score and get the predicted rating values. In some special case, I could not get the predicted rating value, I adopt the average rating value from the training dataset as the default rating score.

```
item.predict.rate <- function(uid, mid, k) {
  item.rate <- data.frame(item.similarity[[mid]][,1], users.movies.mtx[uid, item.similarity[[mid]][,2]])
  item.rate <- item.rate[complete.cases(item.rate),]
  item.rate <- item.rate[order(item.rate[, 1], decreasing = TRUE),]
  if(dim(item.rate)[1] == 0) {
    mean.rate
  } else {
    top.k <- head(item.rate, k)
    if(sum(abs(top.k[,1])) == 0) {
      mean(top.k[,2])
    } else {
      sum(top.k[,1] * top.k[,2])/sum(abs(top.k[,1]))
    }
  }

}
start_time <- proc.time()
item.predict.out<- apply(test.ratings, 1, function(x) {round(item.predict.rate(x[1],x[2],recommend.num)
comparison[1, 2]<- (proc.time() - start_time)[3]
comparison[2, 2] <- rmse(test.ratings$rating, item.predict.out)
item.predict.out<-data.frame(test.ratings[,-c(4)],predict = item.predict.out)
item.precision <-item.predict.out %>% group_by(userid)%>% arrange(desc(predict)) %>% slice(1:recommend.
item.recall<-item.predict.out %>% group_by(userid)%>%summarise(total.num = sum(rating>=4,na.rm = TRUE))
comparison[3, 2] <- mean(item.precision$good/item.precision$recommend.num, na.rm = TRUE)
comparison[4, 2] <- mean(item.precision$good/item.recall$total.num, na.rm = TRUE)
```

The (process.time, rmse,precison and recall) for the item-based recommendation system is (5.34, 1.02, 0.67, 0.6)

Conclusion

Given this 100K movie dataset, I made the comparison table for those two recommendation system in the following table.

	user.based	item.based
prcoess.time	5.0700000	5.3400000
rmse	1.0829394	1.0232402

	user.based	item.based
precision	0.6799576	0.6661718
recall	0.6072355	0.5962480

As you can see,the precison and recall of the user-based recommendation system is slightly better than item-based recomendation system.Besides that, the user-based processing time is shorter than item-based processing time. The only weekness of the user-based recommendation system is that the RMSE is slighly bigger than item-based recommendation system.

Scenario

From the experiment, it seems the user-based recommendation system is better than item-based recoomendation system. This conclusion is based on the small dataset(100K). As the data is getting bigger, the user-based recommendation system will need more memory to build up user similarity matrix and may take more time to predict. In the big data scenario, the item-based recommendation don't need to keep large similarity matrix and can predict the rate quickly.

Further work

This dataset also include u.user and u.item. u.user includes more user information including gender,age and occupation. u.item includes more movie information include genre. In the user-based recommendation system, we can make use of the u.user and redefine similarity function. If we can consider gender and age information into our similarity function, we can get the higher prediction accuracy. For the same purpose, we also can consider genre attribute in the item-based recommendation system.

Reference

Item-Based Collaborative Filtering Recommendation Algorithms